collect_messages—this routine will read through each input channel to see if a message exists. If one exists, then the message will be read into a temporary buffer and passed to the routine place_message for placement on the input buffer.

place_message—this routine will determine if the message being read in is the first or subsequent message. If the message is the first packet, then the routine parse_long_hdr is called. If the packet is the second or subsequent message, the routine parse—short_hdr is called.

parse_short_hdr—this routine is used to parse the structure tmp_buffer, validate the command being sent along with the correct packet number, then place the data in the appropriate in buffer message structure. This routine is called after the initial command packet is sent.

parse_long_hdr—this routine is used to parse the structure tmp_buffer, validate the command being sent. This routine then initializes the in_buffer structure and finally places the data in the in_buffer messages structure. This routine is called when a handheld device sends its initial packet for requesting a command.

comm_server—this routine is the main routine which controls the logic flow of the entry communication server system.

process_in_buffer—this routine is used to search the in_buffer for completed messages to e placed on the process queue. This routine determines if a messages is finished if msg_total=—msg_len and msg_total >0. This routine calls ENQUEUE to physically put the message on the process queue.

process_cmd—this routine is used to take the next message off the command queue then send it to the correct server for processing. This routine calls the dequeue function to physically dequeue the item off the command queue.

process_transmit—this routine is used to take the next message off the transmit queue then send it to the correct server for processing. This routine calls the dequeue function to physically dequeue the item off the transmit queue.

packet_msg—this routine is used to physically send the message from the communication server to the handheld computer via the AndroDat Card. It is responsible for all packet of the data and verify of successful transmission.

enqueue—this routine takes completed messages and places them on the queue. A parameter is passed to which queue to place the message along with the data to queue.

dequeue—this routine will take messages off the queue and place in a temporary buffer to be processed by the communication server. A parameter is passed to which queue to process along with the temporary space to put message.

nulqueue—this routine is used to initialize a queue and prepare it to start receiving data. Call to this routine will destroy any data existing in the queue.

com_info—this structure is used to store the incoming message from the handheld

in_buffer—this is an array of the structure com_info. This structure is used to hold the individual messages being received from the handheld computers.

message_list—this structure is used to serve as the queuing functions for the command processing and data transmission.

long_hdr—this is the structure of the header for initial communication to the server.

short_hdr—this is the structure for second and subsequent communications of the second command.

What is claimed is:

1. A multi-tiered computing system comprising:

a first computing tier having a master server computer, the master server coupled to a master database of master data records;

a second computing tier having a plurality of input computers, each input computer coupled to a respective local database of local data records, each local database replicated from the master database; and

a third computing tier having a remote portable computer, the portable computer being in communication with a select input computer for accessing local data records, the portable computer comprising:

a CPU for controlling operation of the portable computer;

an interface memory for storing software to control the CPU and to store temporarily at least one data set each data set having multiple data fields for storing data values; and

a touch sensitive display screen for displaying at least a data I/O screen and for sensing contact by a user, the CPU defining multiple virtual regions upon the data I/O screen, each corresponding to a data field, the display screen sensing and informing the CPU of contact by the user within a virtual region, the display screen displaying within each virtual region a data value for the associated data field from a current data set for a current matter, the CPU identifying a virtual region contacted by the user and effecting an interface control associated therewith and wherein the user modifies data values within a correspond active data field by pressing the associated virtual region.

2. The system of claim **1** wherein the input computer comprises:

a command server for managing the local database; and

a communications server for receiving and transmitting packets of information to and from the portable computer, the packets being constructed in a first format having a header and a data segment, the communications server converting the packets to a second format and constructing a message therefrom, the communications server transmitting the message to the command server which returns a message list, the communications server converting the returned message list to the first format and transmitting a packet of information to the portable computer.

3. The system of claim **1**, wherein the display screen illustrates the current data set in at least one of a scroll bar format and a rolling key format.

4. The system of claim **1** wherein the display screen displays one of a set of display screens including the data I/O screen and including a menu screen and a graphing screen, wherein each selection from the menu screen corresponds to a virtual region and an associated processing sequence.

5. The system of claim **1**, wherein each virtual region corresponds to a predefined processing sequence which is initiated by the user by contacting the associated virtual region.

6. The system of claim **1**, wherein the data I/O screen further displays multiple icons, each being uniquely associated with a data field.

7. The system of claim **3**, wherein said rolling key format includes multiple sets of displayed keys, each set corresponding to a unique data field, each displayed key within each set corresponding to a digit within a related data value and to a unique virtual region.

8. The system of claim **7**, wherein the user adjusts a current data value of an activated data field by contacting a virtual region corresponding to a displayed key within the set of displayed keys associated with the active data field.

9. The system of claim **3**, wherein the scrolling bar format includes a displayed scroll bar with a current level corresponding to a data value of an activated data field.